BAB 2 KAJIAN PUSTAKA DAN DASAR TEORI

Kajian pustaka dan dasar teori berutujuan untuk membentuk kerangka teori yang berasal dari *literature* yang berhubungan dengan masalah yang diteliti. Bab ini membahas tinjauan pustaka yang digunakan untuk menunjang penulisan skripsi, dimana dalam tinjauan pustaka terdapat kajian pustaka mengenai referensi dari penelitian sebelumnya yang terkait dengan kelebihan algoritme yang diimplementasikan dalam penelitian. Sedangkan dasar teori membahas mengenai teori-teori dasar yang berhubungan dengan konsep dasar pemilu, sistem keamanan, kriptografi, algoritme Grain, algoritme hash SHA3 dan perangkat bergerak.

2.1 Kajian Pustaka

Pada sub bab ini mmbahas tentang alasan mengapa skripsi ini menggunakan algoritme krptografi Grain dan hash SHA-3 berdasarkan jurnal penelitian terlebih dahulu. Penelitian sebelumnya berjudul "*Energy Eficiency of Encryption for Wireless Device*" (Elminaam, 2009) menyarankan untuk mempertimbangkan penggunaan *resource* batrei pada perangkat *wireless*, mengingat sangat terbatasnya sangat terbatasnya kapasistas batrei pada setiap perangkat. Daya batrei juga bergantung pada masalah energy yang dikonsumsi dikarenakan algoritme kriptografi yang digunakan. Dewasa ini telah adanya muncul suatu sebutan "*Battery Gap*" yang dikarenakan perkembangan baterai lebih lambat dibandingkan dengan perkembangan teknologi lainya, sehingga diperlukan cara untuk memanfaatkan *resource* atauapun kapasistas batrei dengan sebaik-baiknya dan seefisien mungkin. Pada penelitian lainya dijelaskan bahwa Algoritme Grain didesain untuk berjalan pada *hardware* dengan *resource* yang sedikit. Grain juga disebut dapat berjalan pada lingkungan dengan konsumsi power dan memori yang sangat terbatas. Namun Grain juga dapat ditingkatkan kecepatanya jika *resource* dari *hardware* memungkinkan (Hell Martin, Thomas Johanson dan Willi Meier, 2007). Tingkat integritas pada pemilu *online* juga sangat diperlukan sehingga penulis memutuskan untuk menambahkan fungsi hasing menggunakan SHA-3. Pada situs resmi *National Institute of Standards and Technology* (NIST) dikatakan bahwa *Keccak* merupakan pemenang dari kompetisi yang diadakan NIST dan juga dijadikan menjadikan *Keccak standart* untuk SHA-3 pada tahun 2015 menggantikan SHA-1.

2.2 Dasar Teori

Dasar teori membahas tentang materi-materi yang berkaitan dengan pengembangan aplikasi *system* keamanan aplikasi *client* pemilu *online* dengan algoritme kriptografi Grain dan teknik *hashing* SHA3. Hal-hal yang dibahas pada bagian ini adalah pemilu, *system* keamanan, kriptografi, algoritme Grain, algoritme hash SHA-3, platform *android*, aplikasi *client server* dan perangkat bergerak.

2.2.1 Pemilihan Umum

Pemilihan umum atau yang biasa lebih dikenal dengan pemilu adalah wujud nyata demokrasi prosedural, meskipun demokrasi tidak sama dengan pemilihan umum, namun pemilihan umum merupakan salah satu aspek demokrasi yang sangat penting yang juga harus diselenggarakan secara demokratis (Veri Junaidi, 2009).

Pemilu juga merupakan sarana perwujudan kedaulatan rakyat sekaligus merupakan arena kompetisi yang paling adil bagi partai politik, sejauh mana telah melaksanakan fungsi dan perannya serta pertanggung jawaban atas kinerjanya selama ini kepada rakyat yang telah memilihnya (Sukriono Didik, 2009).

2.2.2 Pemilu *Online* (E-Voting)

Definisi e-voting jika dicermati adalah tujuanya yaitu lebih mengacu kepada pemanfaatan perangkat elektronik untuk lebih memudahkan dan melancarkan proses dan mengotomatisasi segala kemungkianan camput tangan individu dalam tiap prosesnya (Smith dan Clark, 2005). Salah satu definisi *e-voting* adalah suatu *system* pemilihan dimana data dicatat, disimpan dan diproses dalam bentuk informasi *digital* (VoteHere Inc, 2002).

E – voting atau *electornic voting* secara sederhana dapat diartikan sebagai penggunaan hak pilih dalam sebuah peilu dengan menggunakan teknologi (secara elektronik) (Darmawan. et al., 2014)

2.3 Sistem dan Sistem Keamanan

Sistem bukan merupakan entitas yang berdiri sendiri, melainkan terdapat dalam suatu lingkugan. Lingkungan ini mempengaruhi fungsi dan kinerja *system*. Kadangkala lingkungan bisa dianggap sebagai *system* pula tetapi lebih umumnya, lingkungan terdiri dari sejumlah *system* lain yang berinteraksi satu dengan lainya (Sommerville Ian, 2003).

Sistem kemanan adalah tindakan pencegahan terhadap segala bentuk penyebab kerugian, termasuk didalamnya kerugian secara fisik dan non fisik, berwujud atau tidak berwujud, serta adanya bermacam-macam kerugian oleh berbagai sebab.

Garfinkel mengemukakan bahwa *system* keamanan melingkupi lima aspek, meliputi (Widiyanto, 2007) :

1. *Privacy /Confidentiality*

   Aspek *privacy* atau *confidentiality* merupakan sebuah usaha untuk menjaga dan memberikan keamanan informasi seseorang agar tidak dapat diketahui oleh pihak lain. *Privacy* lebih kearah data-data yang bersifat rahasia sedangkan *confdentiality* berhubungan dengan data yang diberikan kepada pihak lain dengan maksud dan tujuan tertentu.

2. *Integrity*

   Aspek *integrity* atau integritas merupakan usaha untuk menjaga dan menjamin bahwa suatu data tidak dapat dirubah-rubah tanpa seizin pemilik. Jika terdapat perbedaan maka boleh dibilang aspek integritas tidak tercapai.

3. *Authentication*

   Aspek *authentication* adalah aspek yang menyataka bahwa sebuah informasi benar-benar asli, jika berhubungan dengan subjek berarti subjek tersebutlah yang benar-benar dimaksut.

4. *Availability*

   Aspek *Availability* ialah jaminan bahwa data atau informasi yang dimaksut benar-benar ada atau tersedia.

5. *Access Control*

   Aspek *Access Control* ialah aspek yang mengatur hak akses sebuah informasi agar sebuah informasi yang bersifat rahasia tidak dapat di akses orang yang tidak memiliki kepentingan. Misalnya, seorang *administrator* memiliki hak akses penuh terhadap sebuah *computer*, tetapi hal ini tidak berlaku bagi *account guest* ataupun *limited account* lainya.

2.4 Kriptografi

Kriptografi diambil dari Bahasa Yunani yaitu "*crypthos*" dan "*graphein"*. *Crypto* berarti rahasia dan *graphein* berarti tulisan. Sehingga kriptografi memiliki arti yaitu tulisan rahasia. Kriptografi merupakan bidang ilmu tentang teknik enkripsi dimana data sengaja diacak menggunakan suatu algoritme enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Kegiatan dekripsi hanya dapat dilakukan dengan menggunakan kunci dekripsi untuk mendapatkan kembali data asli (Kromodimoeljo, 2009).

Kriptografi sendiri mempunyai komponen-komponen untuk mencapai tujuan kriptografi. Menurut (Ariyus, 2009), beberapa komponen dalam kriptografi meliputi:

1. Enrkipsi (*Encryption*)

   Enkripsi merupakan hal yang sangat penting dalam kriptografi untuk mengamankan sebuah informasi agar pesan yang dikirimkan terjaga kerasahasiaanya. Informasi (yang disebut *plain text*) diubah menjadi serangkaian kode rumit yang sulit diartikan. Enkripsi sendiri bisa diartikan sebagai *chipper* atau kode. Berdasarkan ISO 7498-2, *terminology* yang lebih tepat digunakan untuk menamakan proses ini adalah "*enchiper*".

2. Dekripsi (*Decryption*)

   Dekripsi merupakan kebalikan dari proses enkripsi. Dekripsi yaitu proses mengubah kembali pesan yang telah di enkripsi menjadi pesan

aslinya, yang disebut dengan dekripsi pesan. Berdasarkan ISO 7498-2, *terminology* yang lebih tepat untuk menamakan proses ini adalah "*dechiper*".

3. Kunci (*key*)

   Kunci yang dimaksud disini adalah kunci atau sandi yang digunakan untuk melakukan enkripsi maupun dekripsi. Kunci terbagi menjadi dua bagian, yaitu kunci *private* (*private key*) dan kunci *public* (*public key*).

4. *Plain text*

   Plain text disebut juga *cleartext*, yaitu pesan asli yang ditulis atau diketik. *Plain text* inilah yang akan diproses menggunakan algoritme kriptografi agar menjadi *Chiper text.*

5. Pesan

   Pesan bisa berupa data atau informasi yang dikirimkan (melalui kurir, saluran komunikasi data dan sebagainya) atau yang disimpan didalam media penyimpanan (kertas, *storage* dan sebagainya).

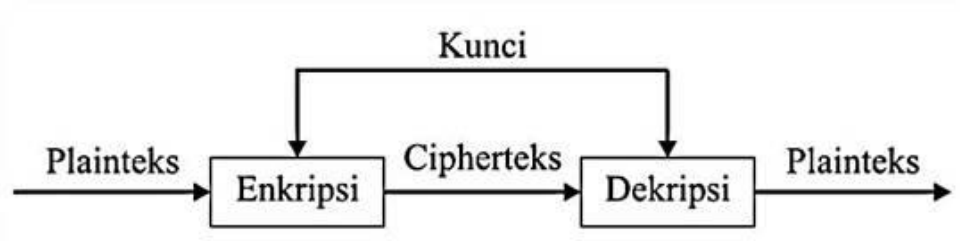
6. *Chiper text*

   Chiper text merupakan pesan yang dihasilkan dari proses enkripsi. Pesan yang terkandung dalam *Chiper text* ini silit dibaca karena berisi berbagai macam karakter tanpa arti/tidak bermakna.

7. Kriptanalisis (*Cryptanalysis*)

   Dapat diartikan sebagai analisis sandi atau suatu ilmu memecahkan *Chiper text* menjadi *plain text* tanpa mengetahui kunci yang digunakan. Pelakunya disebut *cryptanalys* (kriptanalis).

Kriptografi mempunya dua komponen utama yaitu enkripsi dan dekripsi. Selain itu juga dibutuhkan kunci untuk mengubah *plain text* menjadi *Chiper text*, begitu juga sebaliknya. Tanpa kunci, *plain text* tidak bisa melakukan enkripsi pesan menjadi *Chiper text*, juga sebaliknya. Kerahasiaan kunci ini sangatlah penting, apabila kerahasiaanya terbongkar maka isi pesan akan terbongkar. Berikut adalah skema ilustrasi proses enkripsi dan dekripsi.



Gambar 2. 1 Skema Proses Enkripsi dan Dekripsi

Pada gambar 2.4 mengilustrasikan sebuah pesan/*plain text* di enkripsi menggunakan kunci enkripsi sehingga menjadi *Chiper text* yang akan di dekripsi menggunakan kunci dekripsi untuk menghasilkan *plain text* kembali.

2.5 Algoritme Grain

Algoritme Grain adalah algoritme yang sengaja didesain atau diciptakan untuk diterapkan pada *hardware* dengan *resource* yang kecil namun dengan implementasi pada sisi *software* yang mudah. Algoritme Grain didesain agar berjalan dengan berorientasi pada setiap bit bukan setiap kata, hal ini bertujuan untuk menurunkan tingkat kompleksitas namun tetap tangguh dalam menangani masalah keamanan data. Algoritme Grain juga memberikan kebebasan pada penggunanya untuk memiliih tingkat kecepatan komputasi tergantung dari *resource hardware* yang tersedia. Walaupun Grain diciptakan untuk fokus kepada *hardware* namun grain tetap dapat berjalan baik pada *software* pada umumnya, namun grain tidak dapat dibandingkan dengan algoritme lain yang sengaja dibuat untuk efisiensi pada sisi *software*. Grain diciptakan dengan 2 model *shift register* yaitu *linear feedback* (LSFR) dan *nonlinear feedback* (NFSR). Kedua *register* memiliki ukuran 80 bits (Hell, et al., 2007)

Algoritme Grain memiliki 3 blok utama yang dinamakan LSFR, NFSR dan blok *filter*. Sebelum setiap *keystream* diubah menjadi *cipher*, LSFR adalah *shift register* yang bit masukannya merupakan fungsi *linear* dari state sebelumnya. Pada grain *cipher* isi dari LSFR dapat dinotasikan sebagai berikut :

$$S_{\text{I}}, S_{\text{i+1}}, S_{\text{i+2,....}}, S_{\text{i+78}}, S_{\text{i+79}} \quad (1)$$

Fungsi umpan balik dari LSFR, $f(x)$ adalah fungsi *polinom* berderajat 80 yang dapat didefinisikan sebagai berikut :

$$f(x) = 1 + x^{18} + x^{29} + x^{42} + x^{57} + x^{67} + x^{80} \quad (2)$$

Dan fungsi update dari LSFR adalah sebagai berikut :

$$S_{\text{i+80}} = S_{\text{i+62}} + S_{\text{i+51}} + S_{\text{i+38}} + S_{\text{i+23}} + S_{\text{i+13}} + S_{\text{I}} \quad (3)$$

Sedangkan isi dari NFSR dapat dinotasikan dengan :

$$b_{\text{i}}, b_{\text{i+1,}}, b_{\text{i+2,..,}}, b_{\text{i+78,}}, b_{\text{i+79}} \quad (4)$$

Dan fungsi umpan balik dari NFSR , g(x) dapat didefinisikan sebagai berikut :

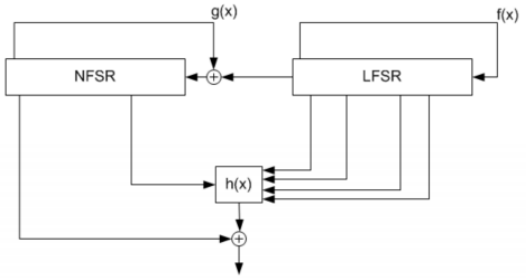
$$g(x) = 1 + x^{17} + x^{20} + x^{28} + x^{35} + x^{43} + x^{47} + x^{52} + x^{59} + x^{65} + x^{71} + x^{80} + x^{17}x^{20} + x^{43}x^{47} + x^{65}x^{71}$$

$$+x^{20}x^{28}x^{35} + x^{47}x^{52}x^{59}$$
$$+x^{17}x^{35}x^{52}x^{71} + x^{20}x^{28}x^{43}x^{47}$$
$$+x^{17}x^{20}x^{59}x^{65}$$
$$+x^{17}x^{20}x^{28}x^{35}x^{43}$$
$$+x^{47}x^{52}x^{59}x^{65}x^{71}$$
$$+x^{28}x^{35}x^{43}x^{47}x^{52}x^{59} \quad (5)$$

Fungsi update dari NFSR dapat didefinisikan sebagai berikut :

$$b_{i+80} = S_i + b_{i63} + b_{i60} + b_{i52} + b_{i45} + b_{i37}$$
$$+ b_{i33} + b_{i23} + b_{i21} + b_{i15} + b_{i9} + b_i$$
$$+ b_{i+63}b_{i+60} + b_{i+37}b_{i+33} + b_{i+15}b_{i+9}$$
$$+ b_{i+60}b_{i+52}b_{i+45} + b_{i+33}b_{i+28}b_{i+21}$$
$$+ b_{i+63}b_{i+45}b_{i+28}b_{i+9} + b_{i+60}b_{i+52}b_{i+37}b_{i+33}$$
$$+ b_{i+63}b_{i+60}b_{i+21}b_{i+15}$$
$$+ b_{i+63}b_{i+60}b_{i+52}b_{i+45}b_{i+37}$$
$$+ b_{i+33}b_{i+28}b_{i+21}b_{i+15}b_{i+9}$$
$$+ b_{i+52}b_{i+45}b_{i+37}b_{i+33}b_{i+28}b_{i+21} \quad (6)$$

Cara kerja dari grain *cipher* secara umum dapat digambarkan dengan diagram sebagai berikut :



Gambar 2. 2 Diagram Umum Grain *Cipher*

Sumber: (Hell Martin,Thomas Johanson dan Willi Meier, 2007)

Isi dari kedua *shift register* disebut sebagai kondisi atau state dari *cipher*. Dari *state* yang didapatkan diambil 5 nilai sebagai masukan untuk fungsi filter h(x). Masukan diambil baik dari LSFR dan NSFR. Fungsi tersebut dapat didefinisikan sebagai berikut :

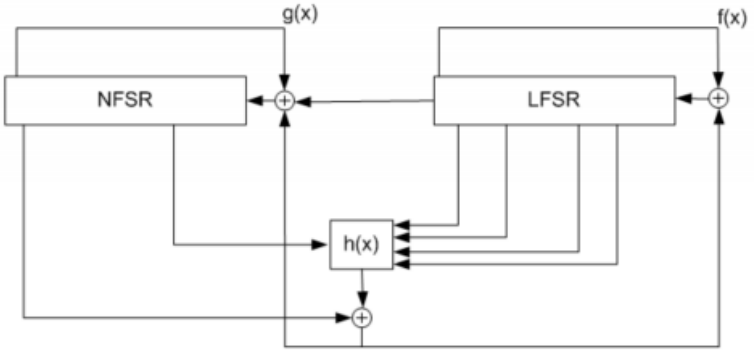
$$h(x) = x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4 \quad (7)$$

Dimana pengubah $\mathcal{X}_0$, $\mathcal{X}_1$, $\mathcal{X}_2$, $\mathcal{X}_3$ dan $\mathcal{X}_4$ diambil dari nilai S_{i+3} , S_{i+25} , S_{i+46} , S_{i+64} , b_{i+63}. Keluaran dari fungsi filter ini di-xor (*masked*) dengan bit bi dari NFSR untuk mendapatkan *keystream*.

2.5.1 Inisialisasi Kunci

Sebelum setiap *keystream* dibangkitkan, *cipher* harus dinisialisasi dengan kunci dan *IV* (Initial Value). Misalkan bit-bit dari kunci, dilambangkan dengan ki dengan $0 \leq i \leq 79$ dan bit-bit dari *IV* dilambangkan dengan *IV*i, dengan $0 \leq i \leq 63$. Inisialisasi kunci dilakukan dengan cara sebagai berikut:

1. NFSR diisi dengan bit-bit kunci, $i = ki, 0 \leq i \leq 79$.
2. 64 bit bertama dari LSFR diisi dengan *IV* , $si = iv$, $0 \leq i \leq 63$.
3. Sisa bit dari LSFR diisi dengan angka 1 (satu) ,$si = 1$, $64 \leq i \leq 79$.
4. Chiper di-Clock 160 kali.Keluaran dari fungsi filter dikembalikan dan di-xor dengan masukan baik ke LSFR dank e NSFR seperti pada gambar dibawah ini



Gambar 2. 3 Inisialisasi Grain *Cipher*

Sumber : (Hell et al, 2007)

2.6 Algoritme SHA3

Pada tahun 2006, NIST mengadakan kompetisi *hash function* untuk membuat sebuah standar *hash* baru, yaitu SHA-3. SHA-3 dibuat tidak untuk menggantikan SHA-2, dikarenakan belum ada serangan hebat yang terjadi pada SHA-2. NIST membuat SHA-3 dikarenakan kekhawatiran dikarenakan MD5, SHA-0, SHA-1 yang telah berhasil ditembus. Karena itulah NIST mencari algoritme *hash* alternatif yang sangat berbeda dengan algoritme sebelumnya, yaitu SHA-3.

Tahun 2012, *Keccak* menjadi pemenang dalam kompetisi ini. Kemudian pada tahun 2015 mempublikasikan draft FIPS 202 tentang *"SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions"*. Setelah itu pada tahun 2015 SHA-3 diresmikan sebagai standar baru fungsi hash (Charles, 2015).

Tidak seperti SHA-1, SHA-3 merupakan kelompok dari *spoge function* dikarenakan memiliki ukuran keluaran yang beragam. Jenis-jenis keluaran SHA-3 tersebut yaitu SHA-3 224, SHA-3 256, SHA-3 384, SHA-3 512, SHAKE128 dan SHAKE256. Menurut pembuat dari algoritme SHA-3 ini, pada jurnalnya yang berjudul *"Sponge function", sponge function* menyediakan cara tertentu agar dapat menggeneralisasikan fungsi *hash* dengan hasil keluaran yang beragam (Bertoni, 2011).

Berdasarkan penjelasan dari publikasi resmi FIPS yang berjudul "*SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*", ada beberapa langkah yang harus dilakukan agar menghasilkan hash SHA-3. Langkah-langkah tersebut akan dijelaskan sebagai berikut.

a. Inisialisasi *input* dan *output*
   Pada fase ini dilakukan pemilihan jenis algoritme SHA-3 mana yang akan digunakan. Jenis-jenis algoritme SHA-3 tersebut yaitu SHA-3 224, SHA-3 256, SHA-3 384, SHA-3 512, SHAKE128 dan SHAKE256. Parameter yang dibutuhkan dari fase ini yaitu *message* masukan dari pengguna, misalnya "plainttext". Masukan tersebut akan diproses, setelah diproses maka akan dihasilkanlah *digest* dari *message* tadi.
b. *Absorbing*
   Pada fase ini, *block state* dari *plain text* yang telah diproses pada fase sebelumnya tadi akan dimasukkan kedalam algoritme dan kemudian diproses lagi. Proses ini juga melibatkan fungsi permutasi Keccak.
c. *Squeezing*
   Fase ini merupakan fase proses perhitungan untuk *digest* yang akan dihasilkan. Panjang yang dihasilkan akan sesuai dengan yang dipilih pengguna pada fase inisialisasi. Fase ini juga melibatkan permutasi Keccak.
d. Permutasi Keccak
   Permutasi Keccak merupakan proses paling utama dari algoritme ini. Semua fase yang dilalui harus melalui fungsi ini dulu. Permutasi Keccak dibagi menjadi 5 tahapan, yaitu ρ *(Rho),* θ *(Theta)*, χ *(Chi)*, π *(Phi)*, dan ι *(Iota)*. Tahapan-tahapan tersebut memiliki fungsi masing-masing, dan fungsi tadi telah ditetapkan oleh NIST. Fungsi-fungsi ini dilihat merujuk dari publikasi resmi FIPS 202 oleh NIST.
   Setiap tahapan akan dilakukan perulangan sebanyak 25 kali.

θ *(Theta)*

Operasi *Theta* bersifat linier. Operasi *Theta* dapat digambarkan sebagai penambahan pada setiap bit A[X][Y][Z] pada penjumlahan bit dari A[X-1][.][Z] dan A[X+1][.][Z-1]. Operasi *Theta* hanya meng-XOR-kan 11 bit menjadi satu, sehingga setiap bit berpengaruh pada 11 bit lainnya. Operasi yang dilakukan pada Theta dijelaskan lewat *pseudocode* sebagai berikut.

$$C[x] = A[x,0] \text{ XOR } A[x,1] \text{ XOR } A[x,2] \text{ XOR } A[x,3] \text{ XOR } A[x,4] \text{ , } x = 0,1,2,3,4 \quad (8)$$

$$D[x] = C[x-1] \text{ XOR } rot(C[x+1],1) \text{ , } x = 0,1,2,3,4 \quad (9)$$

$$A[x,y] = A[x,y] \text{ XOR } D[x] \text{ , } x,y = 0,1,2,3,4 \quad (10)$$

ρ *(Rho)* dan π *(Phi)*

Pada operasi ini akan dilakukan pemetaan terhadap *list array* tadi. Operasi ini juga bersifat linier, dengan invers berupa penggeseran ulang yang berkebalikan dengan penggeseran sebelumnya. Operasi tersebut dapat digambarkan dengan *pseudocode* dibawah ini.

$$B[y,2x+3y] = rot(A[x,y], r[x,y]) \text{ , } x,y = 0,1,2,3,4$$

χ *(Chi)*

Operasi *Chi* merupakan satu-satunya operasi pada permutasi Keccak yang pemetaannya tidak linier. Operasi dapat dilihat sebagai aplikasi operasi *S-box* untuk 5-bit baris. Pada operasi ini dilakukan manipulasi terhadap *list array* B dari operasi sebelumnya yaitu *Rho* dan *Phi*. Kemudian hasil dari manipulasi tersebut dikembalikan lagi ke list array A. *Pseudocode* untuk operasi ini adalah sebagai berikut:

$$A[x,y] = B[x,y] \text{XOR } ((\bar{}B[x+1,y])\text{^}B[x+2,y]) \text{ , } x,y = 0,1,2,3,4 \quad (11)$$

ι *(Iota)*

Dalam operasi ini dilakukan penambahan konstanta ke *list array* pada lokasi [0,0] dari list array A. *Pseudocode* dari operasi tersebut yaitu sebagai berikut:

$$A[0,0] = A[0,0] \text{XOR } RC[i]$$

2.7 Perangkat Bergerak

Perkembangan terknologi dari tahun ke tahun memperlihatkan suatu peningkatan yang signifikan, dimana saat ini teknologi berkembang menjadi suatu perangkat perangkat yang lebih ringkas ataupun lebih kecil daripada sebelumnya, namun memiliki fungsi yang sama, bahkan bisa lebih canggih daripada sebelumnya. Perangkat tersebut biasa disebut sebagai perangkat bergerak, karena kita bisa membawa dan menggunakan perangkat tersebut dimananpun dan kemanapun yang kita ingingkan

Perangkat bergerak merupakan sebuah perangkat kecil yang kemampuan komputasinya terbatas karena beberapa faktor. Perangkat bergerak biasa disebut sebagai perangkat genggam karena kemudahanya dibawa kemana-mana. Perangkat bergerak pada umumnya terdiri dari dua bagian, yang pertama layar *display* sebagai perangkat keluaran dan *key*board atau layar sentuh sebagai masukan. Pada perangkat telepon pintar (*Smartphone*), umumnya perangkat layar visual selain berfungsi sebagai perangkat keluaran, juga berfungsi sebagai perangkat masukan, karena sifatnya yang memiliki layar sentuh pada keseluruhan tampilanya. (Zaki, 2008)

Karena fungsinya yang bervariasi, maka sebuah organisasi T3B dan *DuPont Global Mobility* merumuskan defisini standar dari perangkat bergerak (Zaki,2008), yaitu :

1. Limited Data *Mobile Device* : Perangkat yang ukuran layarnya kecil, umumnya layar hanya menampilkan teks, layanan data hanya terbatas ke SMS dan WAP. Contoh perangkat ini adalah telepon seluler.
2. Basic Data *Mobile Device* : Perangkat yang ukuran layarnya menengah . Memiliki navigasi menggunakan menu atau ikon. Layanan yang ditawarkan antara lain surel (email), daftar alamat (Kontak), SMS, dan *Web Browser*. Contoh perangkat ini adalah telepon pintar (*Smartphone*)
3. Enhanced Data *Mobile Device* : Perangkat yang ukuran layarnya besar, biasanya menggunakan pena stylus untuk layar sentuhnya, dan memiliki fitur-fitur layanan dasar yang dimiliki oleh piranti sebelumnya ditambah adanya kemampuan untuk menambah berbagai aplikasi seperti *Microsoft Office* dan portal internet. Contoh perangkat ini adalah *Pocket PC* dan Tablet.

2.7.1 Aplikasi Perangkat Bergerak

Aplikasi perangkat bergerak (*mobile*) adalah suatu aplikasi yang dibuat secara khusus untuk berjalan pada *mobile device*. Aplikasi *mobile* pada umumnya dikelompokan berdasarkan platform, beberapa kategori platform tersebut adalah:

- *Android*
- iPhone
- Windows *Mobile*
- Blackberry

Aplikasi *mobile* sendiri secara spesifik juga dikembangkan berdasarkan masing-masing platform sesuai dengan kebutuhanya (Native, 2014)

1. *Native Application*

   Aplikasi yang dibuat dan dipasangkan langsung ke dalam *device* menggunakan bahasa pemrograman khusus untuk membuat aplikasi tersebut. Contohnya adalah untuk membuat aplikasi pada platform *Android* digunakan Java dan *Software Development kit* (SDK), juga platform iOS yang menggunakan *Objective*-C dan SDK iOS
2. *Mobile* Web Application

   Mobile Web Application merupakan aplikasi yang membutuhkan browser untuk menjalankanya, biasanya dibuat menggunakan bahasa pemrograman web seperti PHP & HTML 5.
3. Hybrid Application

   Hybrid Application meruapakan gabungan dari aplikasi native dan juga web. Dapat dibuat menggunakan bahasa pemrograman web yang digabung dengan bahasa pemgorgraman yang digunakan untuk membuat aplikasi pada *device* yang dituju.

2.8 *Client Server*

Client Server merupakan salah satu arsitektur dalam jaringan komputer. Arsitektur ini merupakan model konektivitas pada jaringan yang mengenal adanya device yang disebut *server* dan disebut sebagai *client*, dimana masing-masing memiliki fungsi yang berbeda satu sama lain. Prinsip kerja *client server* sangat sederhana yaitu *client* dapat memberikan *request* kepada *server* dan *server* akan menangani *request* tersebut lalu memberikan hasilnya kepada client. *Server* hanya akan bekerja ketika mendapatkan *request* dari *client*.

2.8.1 Sistem *Client Server*

Seperti namanya Sistem *Client* dan *Server* terdiri atas dua komponen (mesin) utama, yaitu *Client* dan *Server*. Setiap aktifitas yang dilakukan oleh *user* umumnya akan lebih dahulu ditangani oleh *client*. *Client* menangani proses yang menjadi tanggung jawabnya. Namun jika terdapat sebuah proses yang harus melibatkan data yang tersimpan pada basis data yang terletak di *server* contohnya, barulah *client* mengadakan hubungan dengan *server* dan memberikan *request* terhadap *server*. *Client* akan megirimkan *request* kepada *server*. Selanjutnya *server* yang menerima *request* tersebut akan menjalankan memproses request *t*ersebut dan hasilnya akan di kirimkan kembali ke *client*. Dengan begitu, transfer datanya jauh lebih efisien.